MThambeliyagodage\_Data605\_W4\_Assign4

Table of Contents

## Problem set 1

In this problem, we’ll verify using R that SVD and Eigenvalues are related as worked out in the weekly module. Given a matrix

A <- matrix(c(1,-1,2,0,3,4),nrow=2)  
A

## [,1] [,2] [,3]  
## [1,] 1 2 3  
## [2,] -1 0 4

* Write code in R to compute

X <- A %\*% t(A)  
X

## [,1] [,2]  
## [1,] 14 11  
## [2,] 11 17

and

Y <- t(A) %\*% A  
Y

## [,1] [,2] [,3]  
## [1,] 2 2 -1  
## [2,] 2 4 6  
## [3,] -1 6 25

* Compute the Eigenvalues and Eigenvectors for X and Y. This will be done with R built-in functions.

# Eigen Values and Vectors for X  
x.val <- eigen(X)$values  
x.val

## [1] 26.601802 4.398198

x.vect <- eigen(X)$vectors  
x.vect

## [,1] [,2]  
## [1,] 0.6576043 -0.7533635  
## [2,] 0.7533635 0.6576043

# Eigen Values and Vectors for Y  
y.eigen <- eigen(Y)  
y.val <- y.eigen$values  
y.val

## [1] 2.660180e+01 4.398198e+00 1.058982e-16

y.vect <-eigen(Y)$vector  
y.vect

## [,1] [,2] [,3]  
## [1,] -0.01856629 -0.6727903 0.7396003  
## [2,] 0.25499937 -0.7184510 -0.6471502  
## [3,] 0.96676296 0.1765824 0.1849001

Then, compute the left-singular, singular values, and right-singular vectors of using the svd command. Examine the two sets of singular vectors and show that they are indeed eigenvectors of and . In addition, the two non-zero eigenvalues (the 3rd value will be very close to zero, if not zero) of both and are the same and are squares of the non-zero singular values of A.

#singular values or diagnals  
A.svd <- svd(A)$d   
A.svd

## [1] 5.157693 2.097188

#left singular  
LS <- svd(A)$u  
LS

## [,1] [,2]  
## [1,] -0.6576043 -0.7533635  
## [2,] -0.7533635 0.6576043

#Multiplying with -1 to aligh with above eign vector x.vect  
LS[,1] <- -LS[,1]  
LS

## [,1] [,2]  
## [1,] 0.6576043 -0.7533635  
## [2,] 0.7533635 0.6576043

#right singular with 3 vectors  
RS <- svd(A, nv = 3)$v  
RS

## [,1] [,2] [,3]  
## [1,] 0.01856629 -0.6727903 -0.7396003  
## [2,] -0.25499937 -0.7184510 0.6471502  
## [3,] -0.96676296 0.1765824 -0.1849001

#Multiplying with -1 to aligh with above eign vector y.vect  
RS[,1] <- -RS[,1]  
RS[,3] <- -RS[,3]  
RS

## [,1] [,2] [,3]  
## [1,] -0.01856629 -0.6727903 0.7396003  
## [2,] 0.25499937 -0.7184510 -0.6471502  
## [3,] 0.96676296 0.1765824 0.1849001

* When tested with all.eqal, I see that two matrices have equal values

all.equal(y.vect, RS)

## [1] TRUE

all.equal(x.vect, LS)

## [1] TRUE

C <- matrix(c(1,2,4,2,-1,3,4,0,1), nrow=3, ncol=3, byrow = TRUE)  
solve(C)

## [,1] [,2] [,3]  
## [1,] -0.02857143 -0.05714286 0.2857143  
## [2,] 0.28571429 -0.42857143 0.1428571  
## [3,] 0.11428571 0.22857143 -0.1428571

## Problem set 2

Using the procedure outlined in section 1 of the weekly handout, write a function to compute the inverse of a well-conditioned full-rank square matrix using co-factors. In order to compute the co-factors, you may use built-in commands to compute the determinant. Your function should have the following signature:

= myinverse()

where is a matrix and B is its inverse and . The off-diagonal elements of should be close to zero, if not zero. Likewise, the diagonal elements should be close to 1, if not 1. Small numerical precision errors are acceptable but the function myinverse should be correct and must use co-factors and determinant of to compute the inverse. Please submit PS1 and PS2 in an R-markdown document with your first initial and last name.

myinverse <- function(A)  
{  
  
### This function will compute the inverse of matrix using co-factors.  
### Input validation will be done for the number of columns and rows for squar matrix.  
### In order to determine the co-factors the built-in determinant function in r will be used.  
  
 row.A = nrow(A)  
 col.A = ncol(A)  
   
 if (NROW(A) == NCOL(A))  
 {   
 if (det(A) == 0) {stop("Determinant is 0. Inverse does not exist")} #check to make sure that determinant is not 0  
 else  
 coeff <- diag(NA, row.A, col.A)  
 for(i in 1:row.A) {  
 for(j in 1:col.A){  
 coeff[i, j] <- (-1)^(i+j) \* det(A[-i, -j]) #looping through for cofactor matrix  
 }  
 }  
 B <- ((t(coeff) / det(A))) #this is from $$A^-1 = C^T/det(A)$$(c is coeff in this case)  
 return(B)  
 }  
 stop('Your matrix is not a well-conditioned or full-rank or square matrix')  
}  
  
  
  
#Testing Function   
C <- matrix(c(1,2,4,2,-1,3,4,0,1), nrow=3, ncol=3, byrow = TRUE)  
C

## [,1] [,2] [,3]  
## [1,] 1 2 4  
## [2,] 2 -1 3  
## [3,] 4 0 1

invC <- myinverse(C)  
invC

## [,1] [,2] [,3]  
## [1,] -0.02857143 -0.05714286 0.2857143  
## [2,] 0.28571429 -0.42857143 0.1428571  
## [3,] 0.11428571 0.22857143 -0.1428571

# Validating my function with the R builtin function solve()  
solve(C)

## [,1] [,2] [,3]  
## [1,] -0.02857143 -0.05714286 0.2857143  
## [2,] 0.28571429 -0.42857143 0.1428571  
## [3,] 0.11428571 0.22857143 -0.1428571